

Free Response: ArcadeCard

Name: _____

Date: _____

An arcade gives each customer a card that holds tokens. Tokens are spent to play games. The arcade numbers its cards in the order they are issued, starting at 1, and keeps a running total of how many tokens have been spent on all cards combined.

The ArcadeCard class has the following behavior.

- A card can be created with an owner's name and a starting number of tokens, or with only an owner's name. A card created with only a name starts with 5 tokens.
- Each card has a card number. The first card created is number 1, the second is number 2, and so on.
- `play(int cost)` attempts to play a game that costs `cost` tokens. If the card has enough tokens, the cost is subtracted, the cost is added to the arcade-wide total of tokens spent, and the method returns `true`. Otherwise the card is unchanged and the method returns `false`. You may assume `cost` is positive.
- `play()` attempts to play a standard game, which costs 2 tokens. It behaves exactly like `play(int cost)` with a cost of 2.
- `addTokens(int amount)` adds `amount` tokens to the card. You may assume `amount` is positive.
- `getTokens()` returns the number of tokens currently on the card.
- `getCardNumber()` returns the card's number.
- `getTotalSpent()` returns the total number of tokens spent on all cards. It can be called without any `ArcadeCard` object, using the class name.

The following table shows a sequence of statements and their results.

Statement	Return value	Explanation
<code>ArcadeCard.getTotalSpent()</code>	0	No tokens have been spent yet.
<code>ArcadeCard c1 = new ArcadeCard("Ana", 20);</code>		c1 has 20 tokens.
<code>ArcadeCard c2 = new ArcadeCard("Ben");</code>		c2 has 5 tokens.
<code>c1.getCardNumber()</code>	1	
<code>c2.getCardNumber()</code>	2	
<code>c1.play(4)</code>	true	c1 now has 16 tokens.
<code>c2.play(8)</code>	false	c2 doesn't have enough tokens and still has 5.
<code>c2.play()</code>	true	Standard game costs 2. c2 now has 3 tokens.
<code>c2.getTokens()</code>	3	
<code>c1.addTokens(10);</code>		c1 now has 26 tokens.
<code>c1.getTokens()</code>	26	
<code>ArcadeCard.getTotalSpent()</code>	6	4 tokens from c1 plus 2 from c2. The failed play spent nothing.

Write the complete ArcadeCard class. Your implementation must meet all specifications and conform to the examples shown in the table.